PTO/SB/33 (07-05)
Approved for use through xx/xx/200x. OMB 0651-00xx
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

PRE-APPEAL BRIEF REQUEST FOR REVIEW

Docket Number (Optional)	
06005/39537	

Application Number	Filed
10/656,005-Conf. #8163	September 5, 2003
First Named Inventor	
Gary K. Law	
Art Unit	**Examiner**
2179	Steven B. Theriault

Applicant requests review of the final rejection in the above-identified application. No amendments are being filed with this request.

This request is being filed with a notice of appeal.

The review is requested for the reason(s) stated on the attached sheet(s).
Note: No more than five (5) pages may be provided.

I am the

☐ applicant /inventor.

☐ assignee of record of the entire interest.
See 37 CFR 3.71. Statement under 37 CFR 3.73(b) is enclosed. (Form PTO/SB/96)

☒ attorney or agent of record.
Registration number 45,127

☐ attorney or agent acting under 37 CFR 1.34.
Registration number if acting under 37 CFR 1.34. _____

/Gregory E. Stanton #45127/
Signature

Gregory E. Stanton
Typed or printed name

(312) 474-6300
Telephone number

October 15, 2007
Date

NOTE: Signatures of all the inventors or assignees of record of the entire interest or their representative(s) are required. Submit multiple forms if more than one signature is required, see below*.

☐ *Total of 1 forms are submitted.

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being transmitted via the Office electronic filing system in accordance with § 1.6(a)(4).

Dated: October 15, 2007 Signature: /Gregory E. Stanton #45127/ (Gregory E. Stanton)

STATEMENT IN SUPPORT OF PRE-APPEAL BRIEF REQUEST FOR REVIEW

The Examiner maintained the rejections of claims 1-79. In particular, claims 1, 15-18, 31-46, 48-50 and 53-55 were rejected as being anticipated by U.S. Patent No. 6,954,724 (hereinafter "Kodosky"). Additionally, claims 2-14, 19-30, 47, 51, 52 and 56-79 were rejected as being unpatentable under 35 U.S.C. §103 over Kodosky in view of one or more of U.S. Patent App. Pub. No. 2002/0194218 (hereinafter "Klapper"), U.S. Patent No. 5,845,063 (hereinafter "Khrapunovich"), and U.S. Patent No. 6,369,836 (hereinafter "Larson"). The rejections should not be upheld at least for reasons best summarized in a discussion of independent claims 1 and 58 provided below. In summary, however, the Examiner failed to establish that the cited references anticipate or render obvious each of claims 1-79.

The embodiments described in the present application are generally related to a function block that implements a state machine, wherein the function block is for use in connection with a process plant. For example, claim 1 is generally directed to a method for configuring such a function block via a computing device having a display device and an input device. Claim 1 recites, *inter alia*, "providing a graphical user interface via the display device for configuring, at least in part, how the state machine is to transition among a plurality of states, wherein the graphical user interface includes a plurality of graphical elements, wherein at least some of the graphical elements can be used to indicate desired transitions between states." Claim 1 also recites, *inter alia*, "receiving state transition data via the graphical user interface; and storing the state transition data on a first computer readable medium associated with the function block."

The final Office Action alleged that Kodosky anticipates claim 1. Kodosky describes a system for generating a hardware implementation of graphical code. The graphical code is exported to a hardware description and a programmable hardware element is configured using the hardware description. For instance, Fig. 10 of Kodosky illustrates a method for exporting a "structure node" into a hardware description. Kodosky explains that a "structure node" is "a node which represents control flow of data" and that an example of a "structure node" is a "While/Do loop[]." *See Kodosky* at col. 18, line 22-31.

Fig. 12 of Kodosky illustrates another method for exporting a structure node into a hardware description when the structure node corresponds to a function block. Additionally, Kodosky provides an example of such a structure node function block. In particular, Fig. 13 of Kodosky is a While loop function block that may be exported into a hardware description. Fig. 14 of Kodosky is a state diagram that illustrates the operation of the While loop function block of Fig. 13. Kodosky does not disclose a method for configuring the function block of Fig. 13 because the system of Kodosky is not concerned with configuring function blocks but rather with converting graphical code into a hardware description. It appears that Kodosky provides Fig. 13 merely to show an example of a function block that can be converted to a hardware description.

The final Office Action failed to establish that Kodosky anticipates claim 1 because the final Office Action failed to show that Kodosky discloses each and every element of claim 1. For example, the final Office Action failed to show that Kodosky discloses or suggests a method "for configuring … a function block associated with a process plant, the function block to implement a state machine, the method comprising: providing a graphical user interface via the display device for configuring, at least in part, how the state machine is to transition among a plurality of states, wherein the graphical user interface includes a plurality of graphical elements, wherein at least some of the graphical elements can be used to indicate desired transitions between states," as recited in claim 1.

The Office Action cited col. 19, lines 49-59 of Kodosky as disclosing a state machine. Thus it appears that the final Office Action alleges that Fig. 13 of Kodosky discloses a state machine. But as discussed above, Kodosky does not disclose a method for configuring the function block of Fig. 13.

The Office Action cited Fig. 16 as disclosing a graphical user interface that includes graphical elements which can be used to indicate desired transitions between states of a state machine. But Fig. 16 of Kodosky does not show a graphical user interface for configuring the function block of Fig. 13. Rather Kodosky states that "FIG. 16 is a conceptual diagram of the resulting hardware after the graphical program example of FIG. 15 is converted into a hardware description." Kodosky at col. 20, lines 31-33 (underlining added). Kodosky states that Fig. 15 "[i]llustrates a simple example of a graphical program" that merely includes two Add function

nodes. Kodosky at col. 20, lines 21-29. Thus, Fig. 16 is not related to Fig. 13. Moreover, Fig. 16 is not related to a state machine. Further, Fig. 16 is not even an illustration of a graphical user interface. Rather Fig. 16 is merely a "conceptual diagram" of hardware after a graphical program is converted to a hardware description. *See Kodosky* at col. 20, lines 31-33.

The Examiner goes on to allege, apparently, that he interprets the FPGA as the state machine of claim 1:

> "The system of Kodosky therefore provides that the interface gives the user the ability to construct a program that manages state of a FPGA that controls an instrument. The FPGA has a series of gates that incorporate states of the machine where inputs to the gates control the output of the machine. With a given input the state of the machine could be on, off, operating, etc."

Final Office Action at p. 37. Moreover, the Examiner apparently alleges that because Kodosky describes a graphical user interface for configuring the FPGA, it therefore recites a graphical user interface for configuring a state machine.

But an FPGA that implements a state machine is not the same as a function block that implements a state machine. As is known to those of ordinary skill in the art, a function block is a sub-unit of operation of a control module that is used for controlling or simulating a process plant. *See e.g.*, *Present Application* at par. [0032]. For instance, Kodosky describes using function nodes, which are icons corresponding to function blocks, to create a graphical program. *See Kodosky* at col. 4, lines 55-64; col. 13, lines 21-31; Fig. 13; col. 19, lines 37-41.

Kodosky gives as an example of a function block a block that implements a While loop. *See Kodosky* at Fig. 13; col. 19, lines 37-41. Kodosky also explains that the While loop function block acts as a state machine. *See Kodosky* at Fig. 14; col. 19, lines 49-59. But Kodosky does not disclose or suggest a graphical user interface for configuring the While loop function block wherein the graphical user interface is for configuring "how the state machine is to transition among a plurality of states, wherein the graphical user interface includes a plurality of graphical elements, wherein at least some of the graphical elements can be used to indicate desired transitions between states" as recited in claim 1. Rather, Kodosky explains that such function blocks are pre-configured:

> "The system of compiling the resulting net list into an FPGA program file preferably uses a library of pre-compiled function blocks to aid in the compilation, as well as hardware target specific information. The library of pre-compiled function blocks includes net list libraries for structure nodes, such as for/next loops, while/do loops, case structures, and sequence structures, among others."

Kodosky at col. 4, lines 55-61 (underlining added). In other words, how the While loop of Fig. 13 transitions among its states is pre-configured.

Claim 1 recites, *inter alia*, a method "for configuring … a function block associated with a process plant, the function block to implement a state machine, the method comprising: providing a graphical user interface via the display device for configuring, at least in part, how the state machine is to transition among a plurality of states, wherein the graphical user interface includes a plurality of graphical elements, wherein at least some of the graphical elements can be used to indicate desired transitions between states." Kodosky does not disclose or suggest these elements of claim 1, among others. At least for these reasons, the rejection of claim 1 should not be upheld.

Claim 58 was rejected as being obvious over Kodosky in view of Klapper. Claim 58 is generally directed to a function block entity for use in a process plant and recites, *inter alia*, "a user modifiable state machine configuration database including state transition data indicative of how a state machine implemented by the function block is to transition among a plurality of states, wherein the state transition data comprises data, for each of the at least some possible pairings of each of at least some of the plurality of states and each of at least some of at least one input to the function block, indicative of a next state to which the state machine should transition when the state machine is in the corresponding state and when the corresponding input is a particular value." None of the cited documents disclose or suggest this element, neither individually nor in combination.

As discussed above, Kodosky does not disclose or suggest configuring a function block that implements a state machine to configure how the state machine is to transition among a plurality of states. Similarly, Kodosky does not disclose or suggest the user modifiable state machine configuration database recited in claim 58.

The final Office Action alleged that Klapper discloses this element. Klapper describes a system for generating logic corresponding to a cause and effects matrix, which can be used to monitor and effect a change in a process system. Fig. 1 of Klapper shows a cause and effects matrix. Rows of the matrix correspond to input variables that are to be monitored, and columns of the matrix correspond to outputs that are to be generated. For example, a row of a cause and effects matrix may correspond to an input variable exceeding a limit. A column may correspond to causing a valve to close. Data displayed at the intersection of a cause and an effect indicates whether the particular cause should result in the particular effect. For example, data at an intersection indicates whether when an input variable exceeds a limit (row) a valve should be closed (column).

The cause and effects matrix of Klapper is not a state transition database as alleged by the final Office Action. Rather, the cause and effects matrix of Klapper merely indicates whether a particular input variable event (e.g., the input variable exceeding a limit) should cause an action (e.g., close a valve). For example, the intersection of column number 1 and row number 1 (Fig. 1) specifies that the action "Close Main Fuel Valve 1" should be taken when "High Furnace Pressure" occurs. It does not specify that a state machine should transition to some next state when the state machine is in a "Main Fuel Valve 1 is Closed" state and when a "High Furnace Pressure" occurs, for example.

Thus, the alleged combination of Kodosky and Klapper does not disclose or suggest the above-discussed elements of claim 58. Moreover, none of the other documents cited in the Office Action disclose or suggest these elements.